REMARKS

The present amendment responds to the Office Action mailed 4 May, 2006.

IN THE CLAIMS

Claims 1-18 were pending on the date of the Office Action. Please amend the claims as provided in the attached new set of claims. The present amendment cancels the pending claims 1-18 and replace them with the new claims 19-30 submitted in the present amendment.

New claims – support in the original disclosure

The new independent claim 19 is based on the former claims 1 and 2. The new claim 19 also includes further clarifying features defining the dynamic behavior of the stack maker module in the constraint enforcer, as explained below:

- The feature indicating that the stack maker module is arranged to perform a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation, is supported by paragraph [0171] in the description.

- The feature indicating that the stack maker module is arranged to perform an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation, is supported by paragraph [0169] in the description.

- The feature indicating that the stack maker module is arranged to perform said delete process followed by said insert process when said stack maker module is called upon as a result of an Update Data Manipulation Language operation, is supported by paragraph [0173] in the description.

The new independent claim 23 is based on the former claims 7 and 8. The new claim 24 also includes further clarifying features defining the performance of the stack maker module, corresponding to the explanation above regarding the new claim 19.

The new independent claim 27 is based on the former claims 13 and 14. The new claim 24 also includes further clarifying features defining the stack maker module in the database system, corresponding to the explanation above regarding the new claim 19.

Dependent claims:

New claim 20 corresponds to former claim 3.
New claim 21 corresponds to former claim 4.
New claim 22 corresponds to former claim 5.
New claim 24 corresponds to former claim 9.
New claim 25 corresponds to former claim 10.
New claim 26 corresponds to former claim 11.
New claim 28 corresponds to former claim 15.
New claim 29 corresponds to former claim 16.
New claim 30 corresponds to former claim 17.

Thus, the new claims 19-30 are considered as being fully supported by the original disclosure.

Patentability of the new independent claim 19

The new independent claim 19 is directed to a transaction based constraint enforcer for a database system, for enforcing a set of constraints that governs the integrity of information stored in the database system. The constraint enforcer is arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction.

Jacobs (col. 1, l. 7.8) relates to the technical field of deferred enforcement of *uniqueness constraints* in a computer system. In relational database terminology, a uniqueness constraint is a prohibition against duplicate values within a column, i.e. it prohibits two or more rows of a table from having the same value in a column or a group of columns. When two rows have the same value in a column or a group of columns, a "uniqueness constraint violation" is said to have occurred. (cf. *Jacobs*, col. 3, l. 4-9). It is respectfully submitted that the present invention relates to a transaction based constraint enforcer that is applicable with *any type* of constraints, such as primary keys, foreign keys, subset constraints, exclude constraints, etc. (cf. par.

[0045]). In contrast, the entire disclosure of *Jacobs* appears to relate to the deferred enforcement of *uniqueness constraints* in particular.

The claimed constraint enforcer of claim 19 further comprises:

- a stack maker module, arranged for creating and updating said check stack, said stack maker module being operatively connected to a runtime module in the database system and arranged to receive data from said runtime module.

Jacobs describes that a list is generated for each uniqueness-required index for each session. An entry is added to the list associated with the index when the counter of an index changes. A transaction may be rolled back to a particular savepoint (col. 9, l. 17-29). The storage for the non-uniqueness count may remain allocated for the duration of a session, i.e. during a connection between the application and the database system (col. 6, l. 48-51). It is respectfully submitted that the above description does not imply that *Jacobs* discloses a stack maker module being operatively connected to a runtime module in the database system and arranged to receive data from the runtime module.

According to the new claim 19, the stack maker module included in the constrain enforcer is further arranged

- to perform a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation,
- to perform an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation, and
- to perform said delete process followed by said insert process when said stack maker module is called upon as a result of an Update Data Manipulation Language operation.

Jacobs apparently fails to disclose at least the above features of the new claim 19.

Accordingly, the new claim 19 more clearly defines the constraint enforcer according to the present invention, in particular the dynamic behavior of the stack maker module included in the constraint enforcer. The new claim 19 clearly points out the stack maker's dynamic behavior as a result of occurring DML operations, and what specified actions that are performed on the check stack in dependency on the calling DML operations. Since the combination of features in

the new claim 19 is not disclosed in or suggested by *Jacobs*, it is respectfully submitted that the new claim 19 is not anticipated by *Jacobs*.

US-5 706 494 *Cochrane* and US-6 453 314 *Chan* both relate to methods for deferred checking of data after *bulk loading* into a database, for checking the database for violation of constraints. A bulk loading phase is an initial population process wherein the database is populated with new data, i.e. a phase in which the records of the database are loaded from an input data set into their target tables.

Cochran and *Chan* apparently fail to disclose a solution for handling database transactions or database manipulating operations *in general*, as it merely mentions constraint checks performed subsequent to a bulk loading phase i.e. a phase which includes INSERT operations only. In contrast, the presently claimed invention is able to handle database transactions in general. Such a transaction is a sequence of DML operations including Insert, Update and Delete operations. The present independent claim 19 includes features that clearly specify how these three types of DML operations are handled by the stack maker module included in the constraint enforcer according to the invention.

Thus, *Cochrane* and *Chan* apparently both fail to disclose at least the features of the new claim 19 that defines the dynamic behavior of the stack maker module when called upon by Insert, Delete or Update DML operations.

US-5,408,657 *Bigelow* relates to a method in a data processing system of imposing constraints on data files which are changed by an update statement. According to the method, the update statement is examined to identify the object of the update and the attribute that is updated. Next, the identified attribute is used to find a multi-object constraint that may be violated by the update. Thereafter the identified object of the update is converted to all objects in the multi-object constraint. Next, it is determined if the constraint is satisfied by all the constraint objects resulting from the converting step. If this determination is false, an error message is generated.

Bigelow apparently fails to disclose at least the step of creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction. *Bigelow* apparently also fails to disclose the further features related to the check stack and the stack maker module of the present invention.

Consequently, the references do not, either alone or in combination, teach or suggest all limitations of the new claim 19.

Patentability of new claims 23 and 27

Analogous reasoning could be carried out with respect to the method of the new independent claim 23 and the system of the new independent claim 27, which contain combinations of features that correspond to the features of the new claim 19. Thus, claims 23 and 27 are also novel and non-obvious with respect to the cited references.

IN THE DRAWINGS /SPECIFICATION

The drawings were objected to for failing to comply with 37CFR 1.84(p)(5) because they included the reference numeral "200" not mentioned in the description. In order to overcome this objection, the applicant has amended the description (p. 6, l. 13, corresponding to paragraph [0047] in the published application) to include reference numeral "200" as indicating the system illustrated in fig. 2. Please amend the fourth paragraph on page 6 of the specification, beginning at line 13 as follows:

> The system 200 illustrated in fig. 2 includes a database runtime module 230. The database runtime module 230 is an executable program, which is generated from a code generator (not shown), which in turn operates according to a data model or schema represented in an offline data dictionary (not shown). The offline data dictionary essentially keeps the same information as would be kept in the online data dictionary 130 shown in fig. 1. The offline data dictionary thus includes meta data including information about constraints.

The drawings were objected to for failing to comply with 37CFR 1.84(p)(5) because they did not include the reference numeral "319" mentioned in the description. In order to overcome this objection, the applicant has amended the description (p. 7, l. 8, corresponding to paragraph [0054] in the published application) to replace reference "319" with reference "310". Please amend the first full paragraph on page 7 of the specification, beginning at line 3, as follows:

> The user 360 is thus allowed to communicate messages to the database system 300. DML statements derived from such messages from the environment, in particular from the user 360, are checked by the constraint enforcer 370 before they are accepted in the database system 300. The constraint enforcer 370 is operatively connected to a conceptual rule unit 380 on the one side, and to the runtime and storage engine ~~319~~ 310 on the other side. The purpose of the constraint enforcer is to assure the 100% principle is satisfied for the database system 300 as a whole.

The drawings were objected to for failing to comply with 37CFR 1.84(p)(5) because reference numerals "310" and "319" were both used to designate "storage engine and runtime module". This objection is remedied by the latter amendment of the description.

ABSTRACT

The specification was objected to because the abstract contained more than 150 words. Please cancel the previous abstract, and substitute the the abstract appear on the attached separate sheet.

Conclusion

In view of the present amendment and the above remarks, allowance of the application is now solicited.